Grafana Labs

From PIDs to Pods: the life cycle of an eBPF-autoinstrumented application

Mario Macias Lloret
Staff Software Engineer

DevBcn

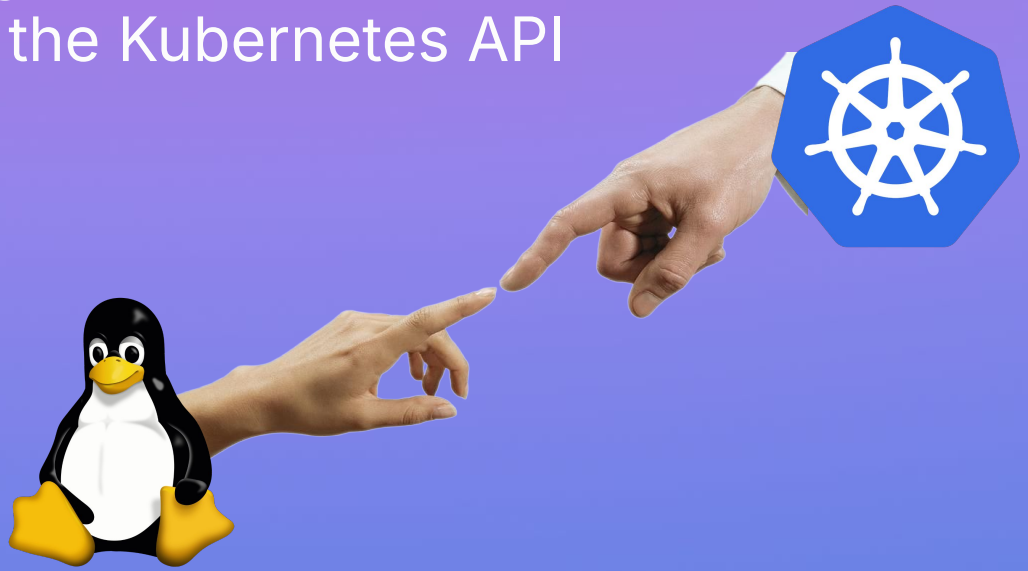
The Barcelona Developers Conference 2025

Grafana Beyla

What's this talk about?

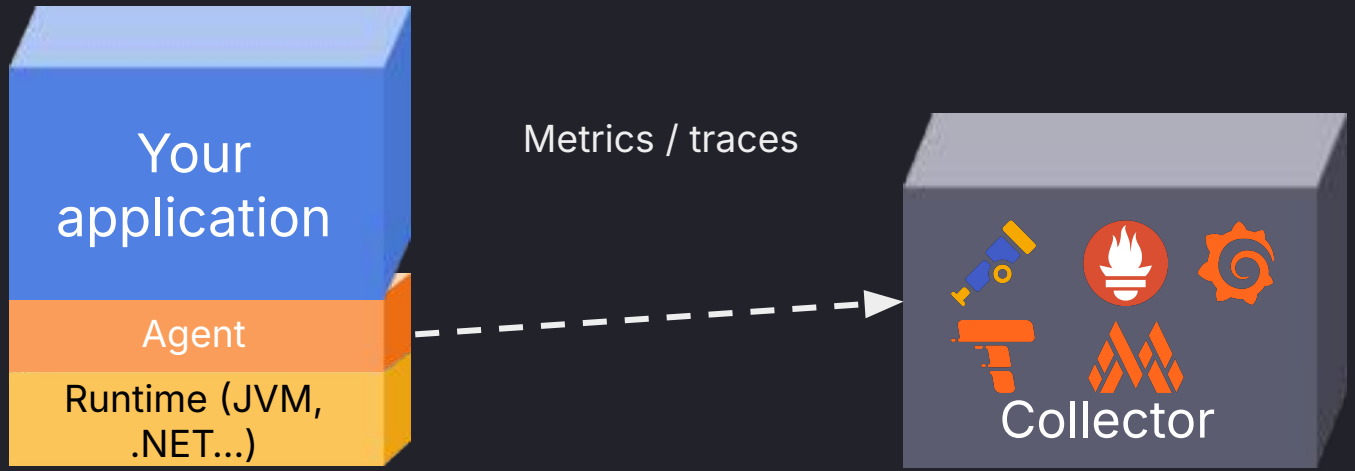
- Introduction to Application Observability
- Introduction to eBPF
- Walkthrough from the internals of the Linux Kernel to the surface of the Kubernetes API

Just for "fun"



Auto-instrumentation with eBPF

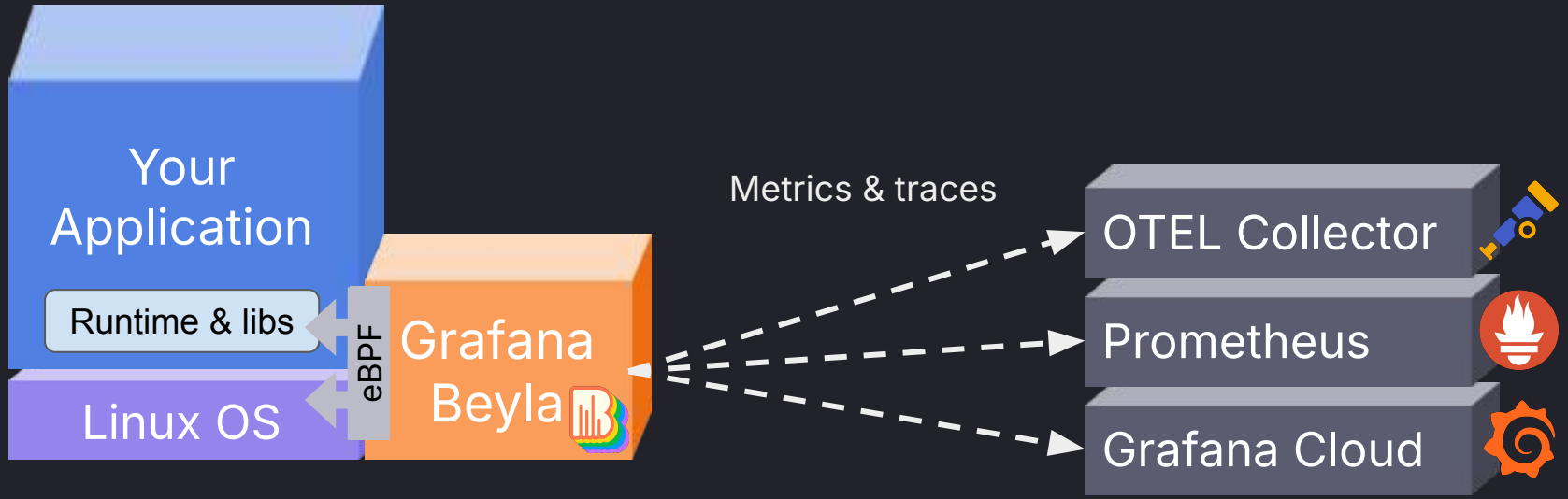
Context: agent-based instrumentation



Agent-based/manual instrumentation: what if...?

- ... my runtime is too old?
- ... too much instrumentation overhead?
- ... my application is a compiled binary?
- ... I don’t want to mess my up code?
- ... I just want instant visibility?

Beyla native eBPF auto-instrumentation



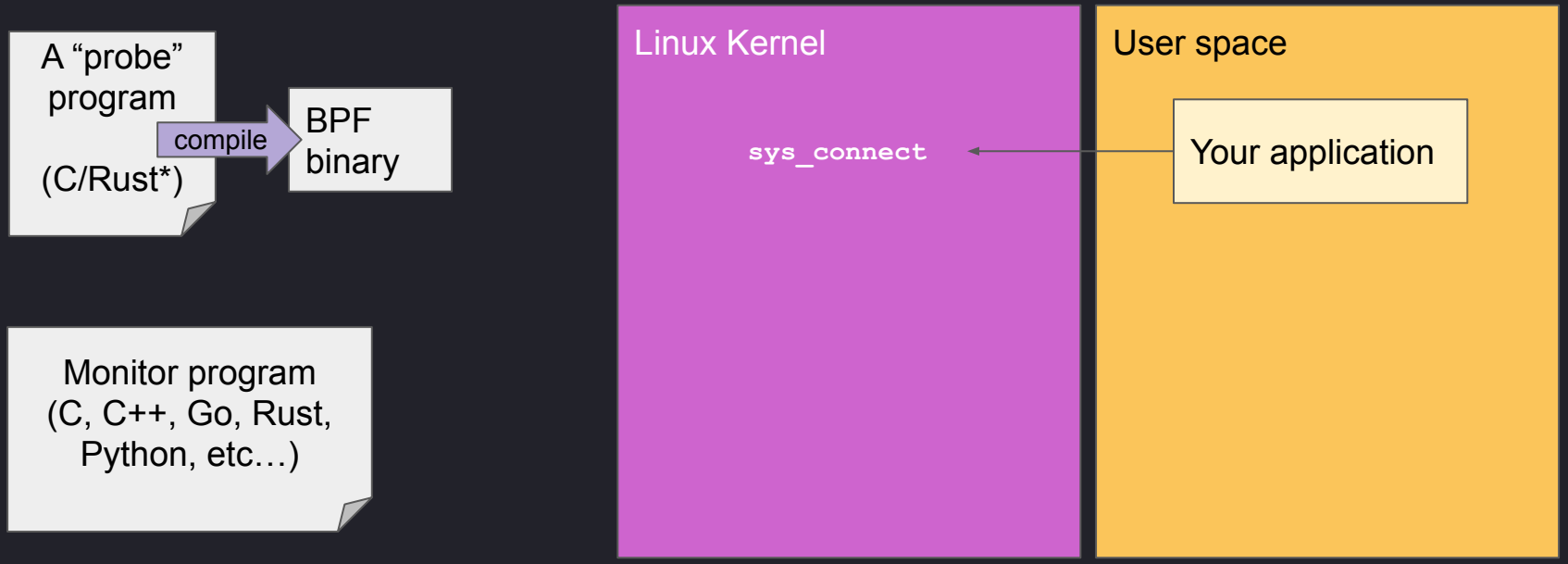
E... B... P... what?

eBPF

- ~~Extended Berkeley Packet Filter~~
  - Virtual Machine built into the Linux Kernel
  - Event-driven programming: “hook” programs into kernel functions and user space programs.
- It requires how the memory is laid out (low-level)
  - Function call arguments
  - Local variables and return values

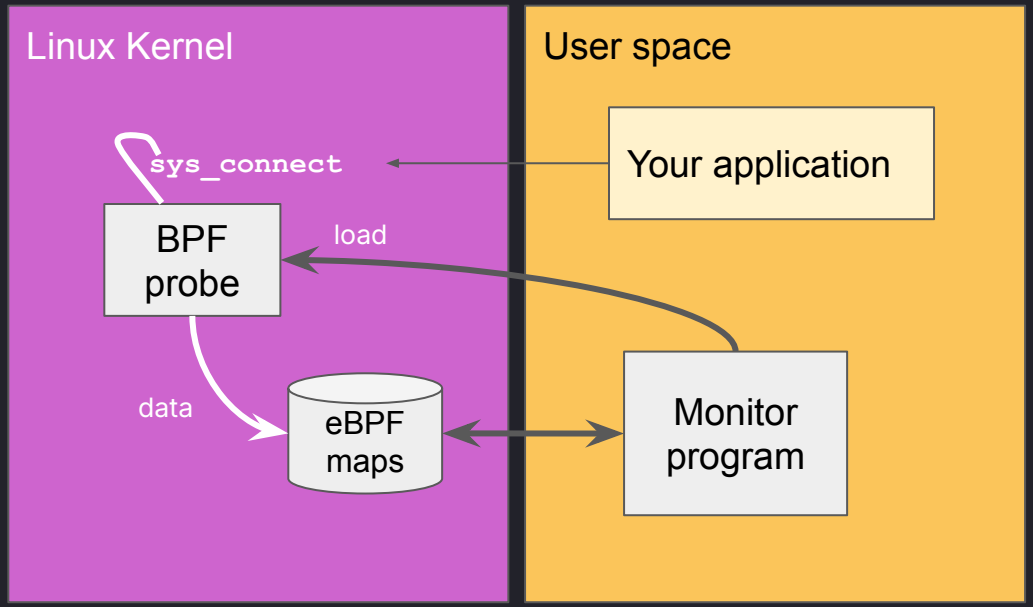
Example: track a new client TCP connection

```
int sys_connect(int fd, struct sockaddr *uservaddr…);
```

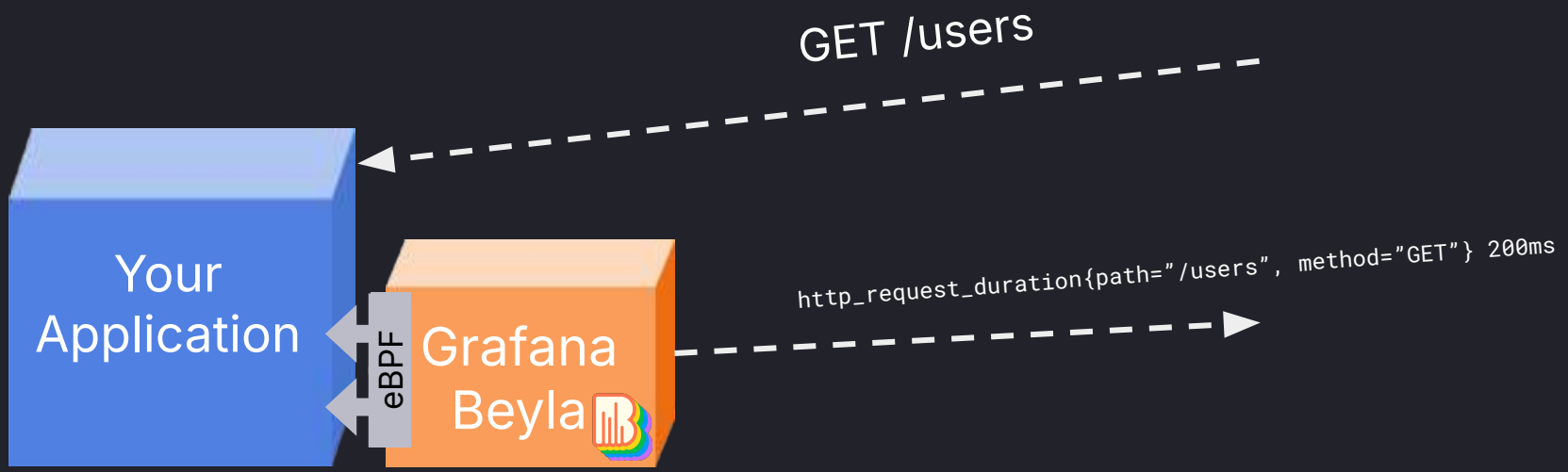


Example: track a new client TCP connection

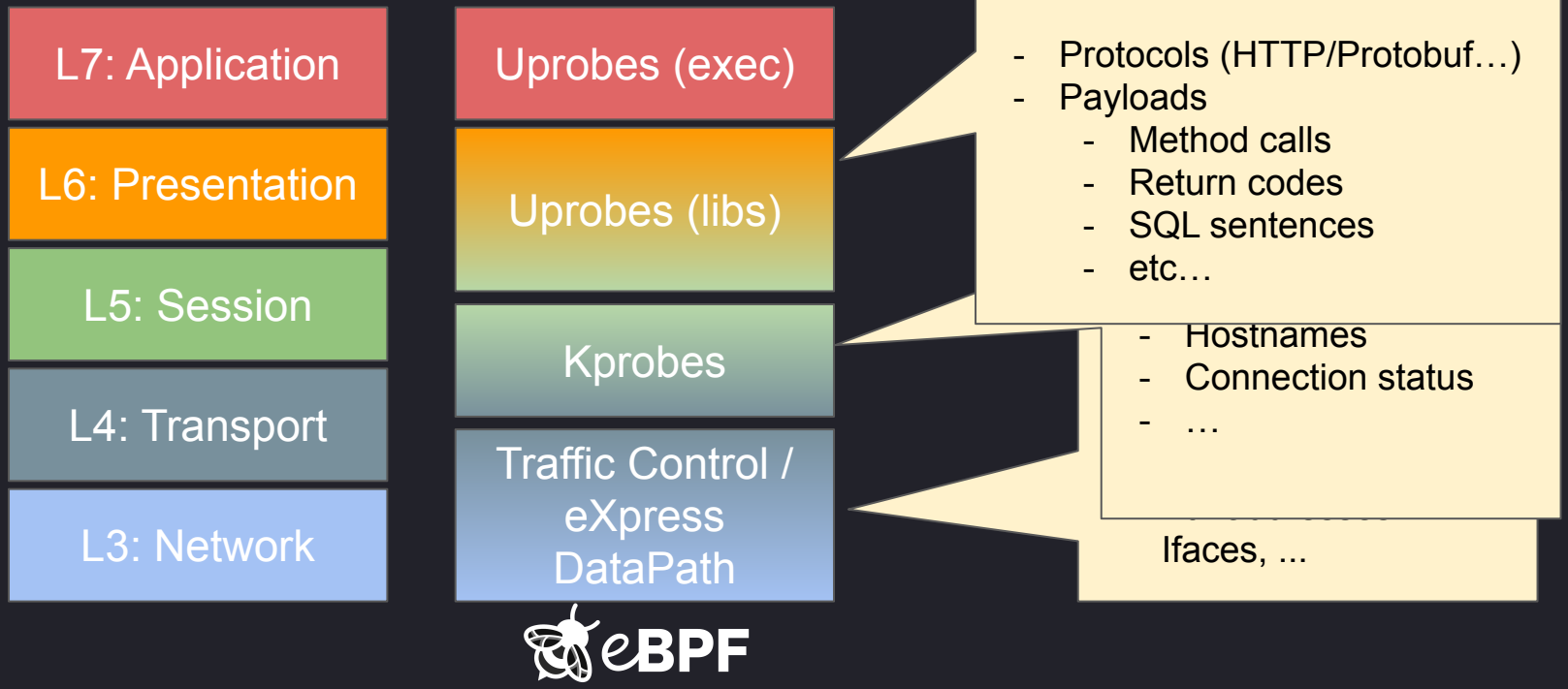
```
int sys_connect(int fd, struct sockaddr *uservaddr…);
```



Observed behavior



How to instrument your network stack



eBPF Pros and Cons

- Pros
  - Fast, JIT compiled probe programs.
  - Safe, all programs are verified at load time by the Kernel.
  - Theoretically, visibility of all your system
- Cons
  - Hard to debug and write.
  - Dependent of the implementation details of the instrumented object.
  - Requires elevated permissions/capabilities.

L3-L4: network-level metrics

- ✌️ Robust: Based on stable APIs and standard binary representations (TCP, UDP, IP... packets)

- 🤷 Basic information
  - Src/Dst endpoints
  - Packet count
  - Bytes sum

L5-L7: application-level metrics / traces

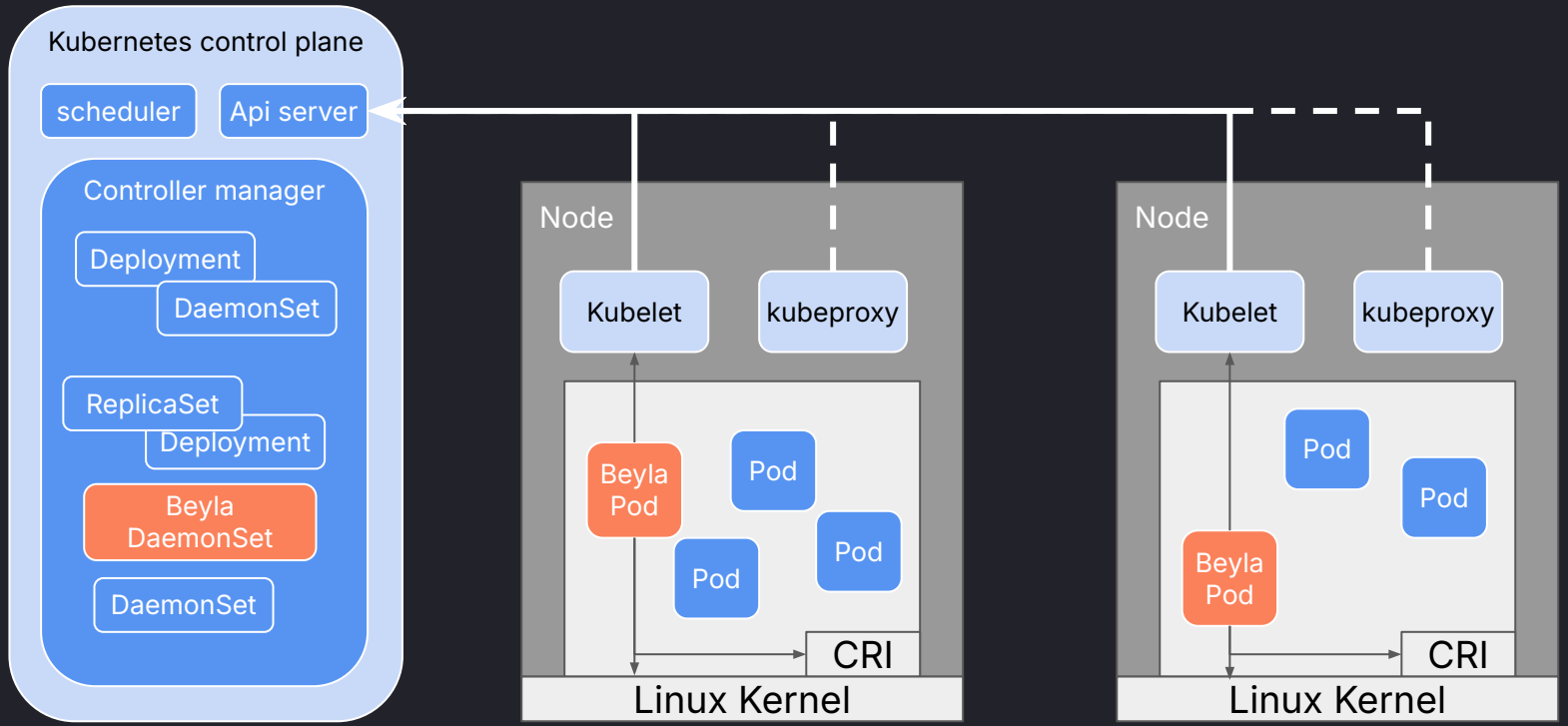
- ✌️ Rich information
  - HTTP/GRPC: methods, response codes, request times...
  - Other protocols: Kafka, Redis, SQL...

- 🤷 Need to implement explicit support to any new implementation of a protocol

- 😭 Relies on internal implementation details that can change with time

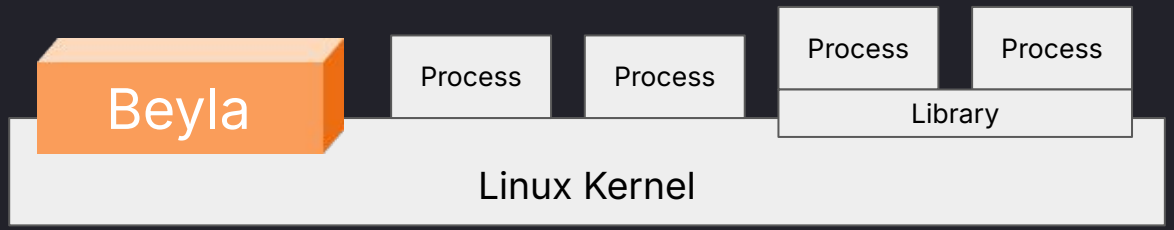
Instrumenting Kubernetes Applications with eBPF

Kubernetes cluster architecture

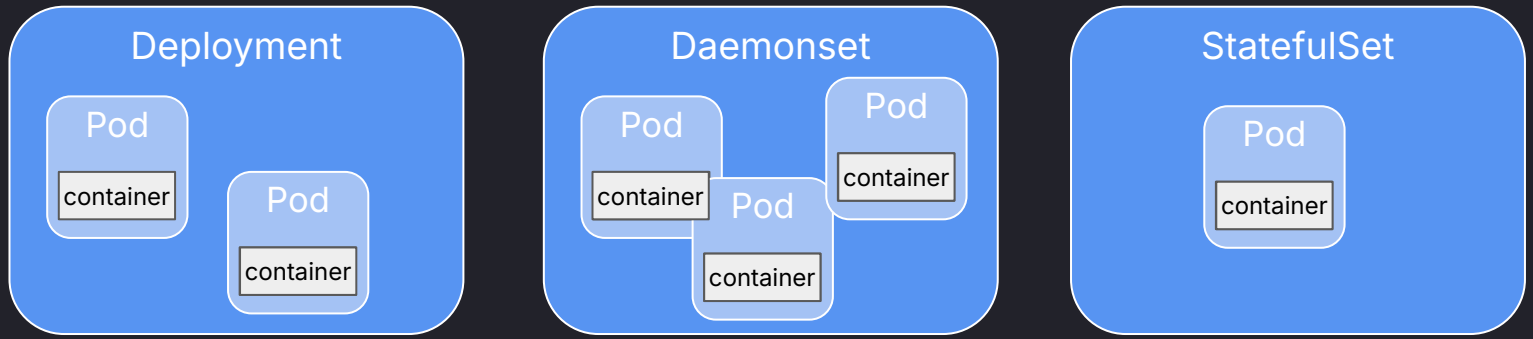


What Beyla directly sees

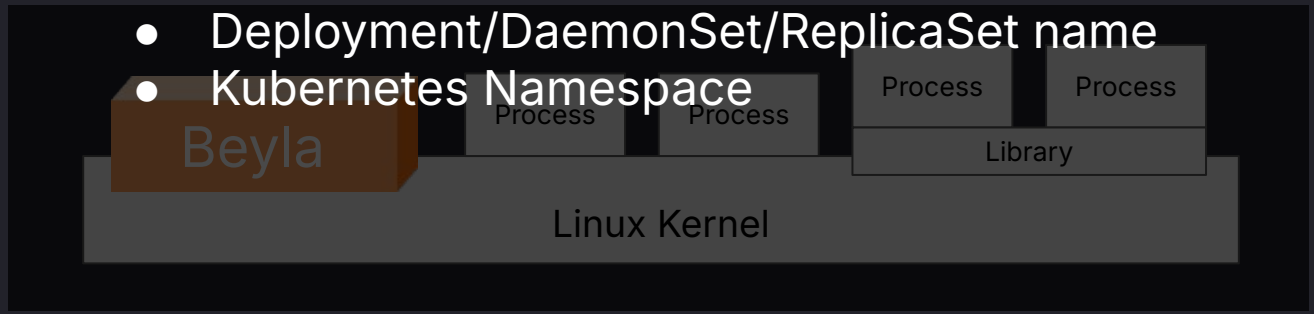
- Command name
- Process ID (e.g. 12145)
- Host Name
- ...



What users actually need



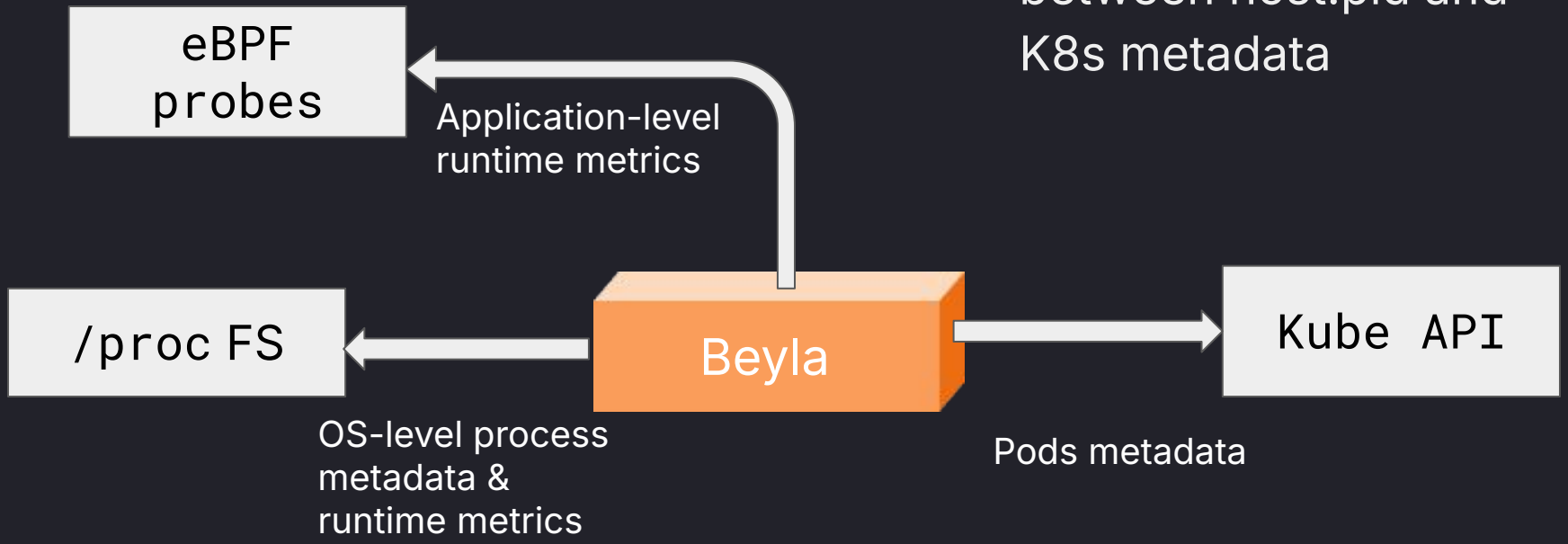
- Pod name & metadata
- Node name
- Deployment/DaemonSet/ReplicaSet name
- Kubernetes Namespace



The Journey from a PID to a Pod

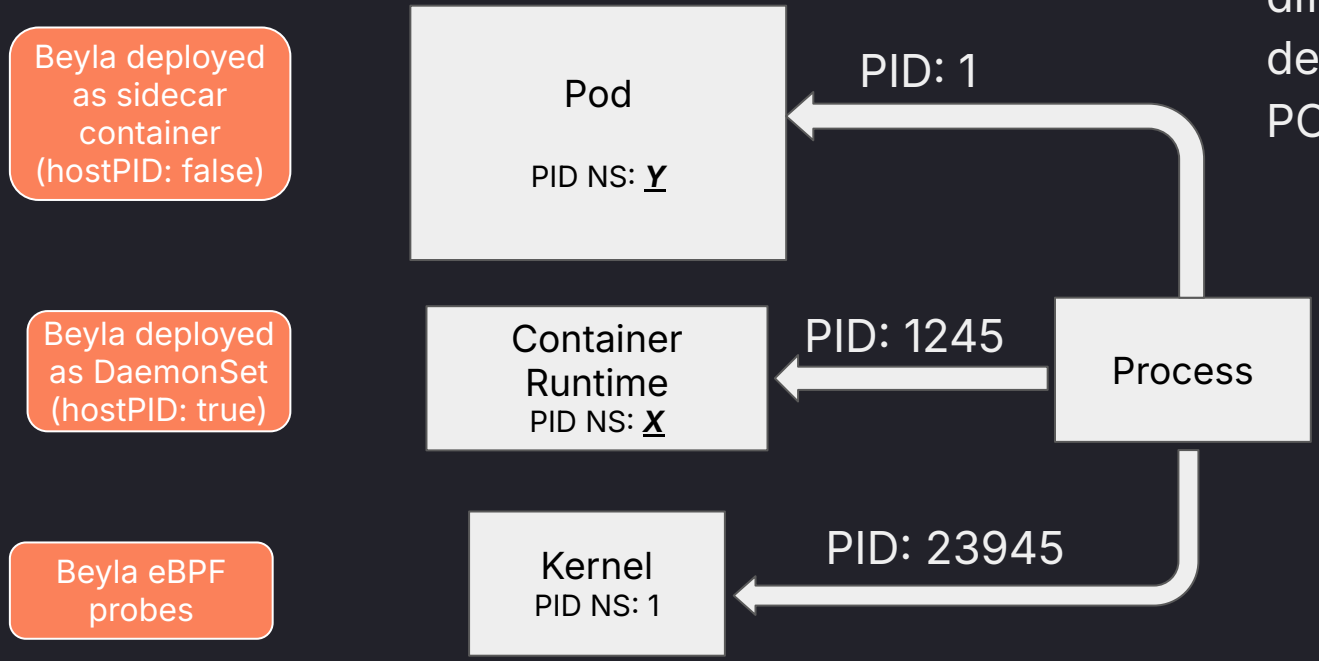
Matching processes with Kubernetes metadata

⚠ No direct mapping between host:pid and K8s metadata

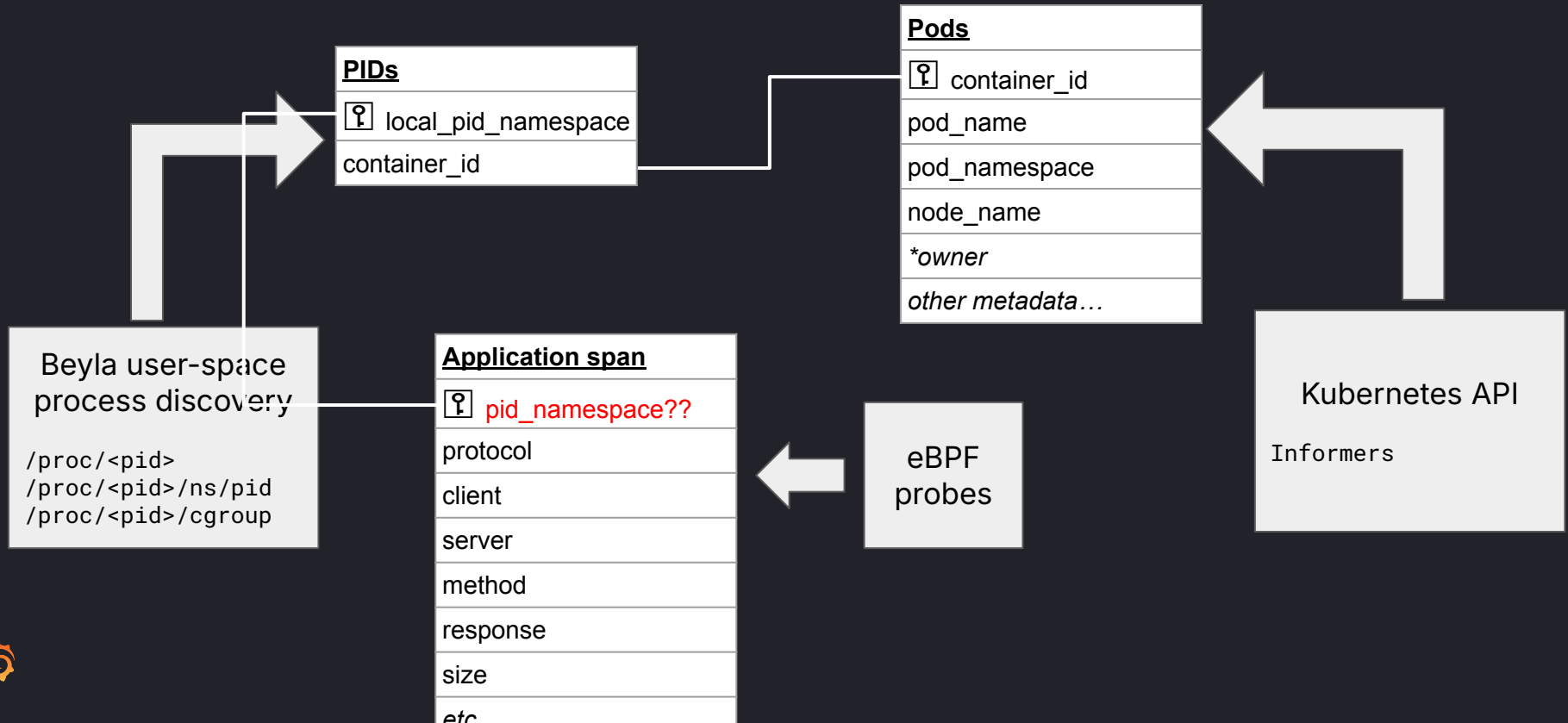


Playing in god mode: PID namespaces

Same process, different PIDs depending on the POV

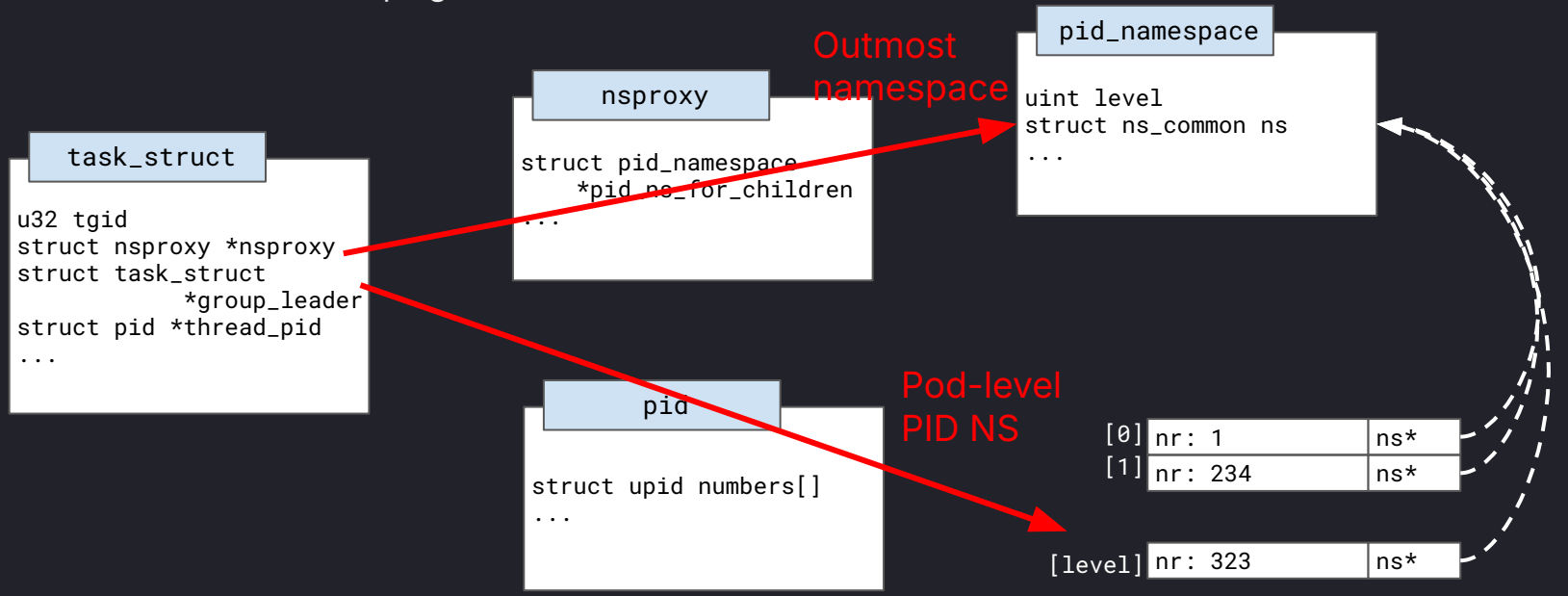


Matching all together

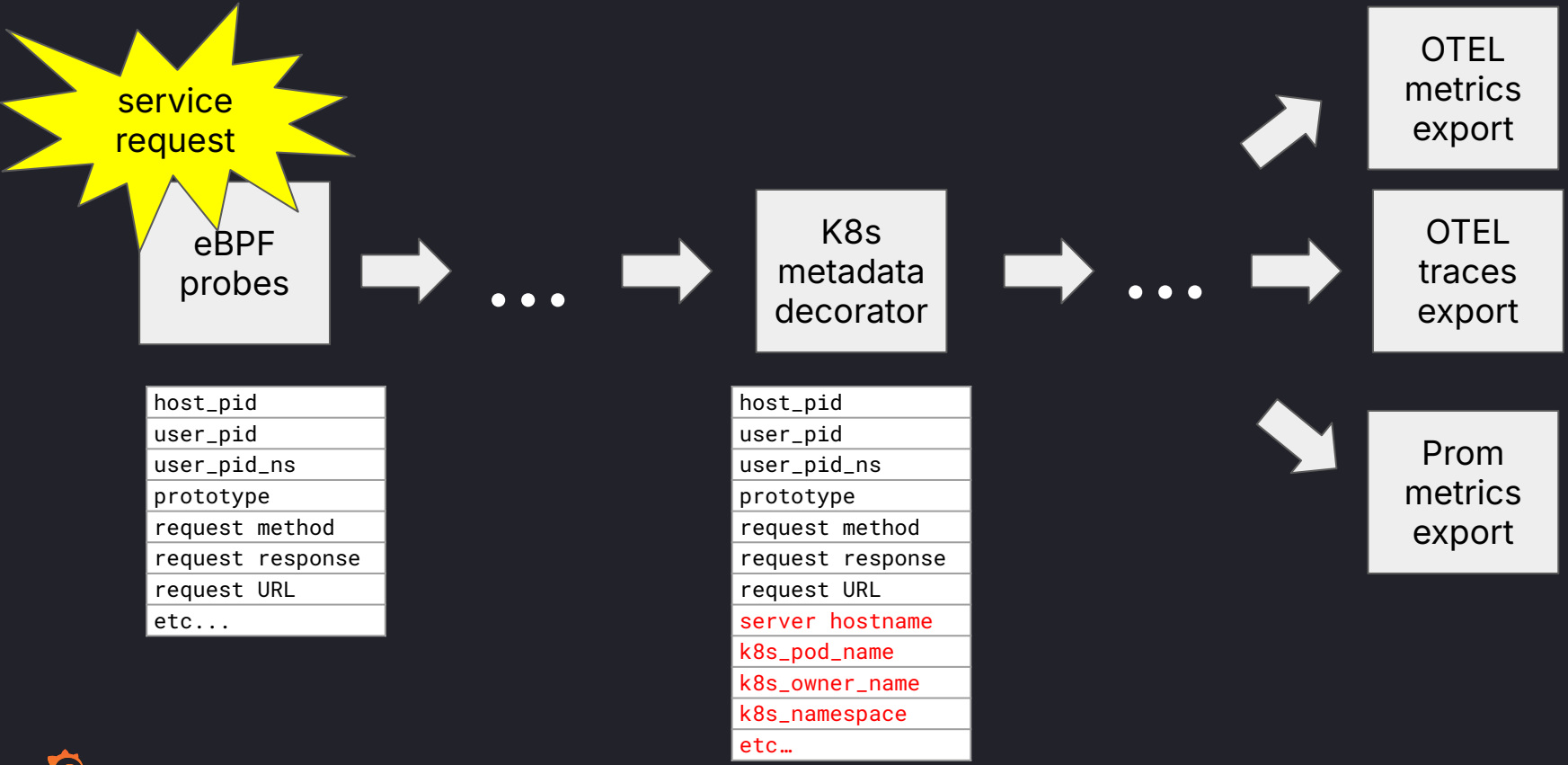


Getting the PID as seen by Beyla

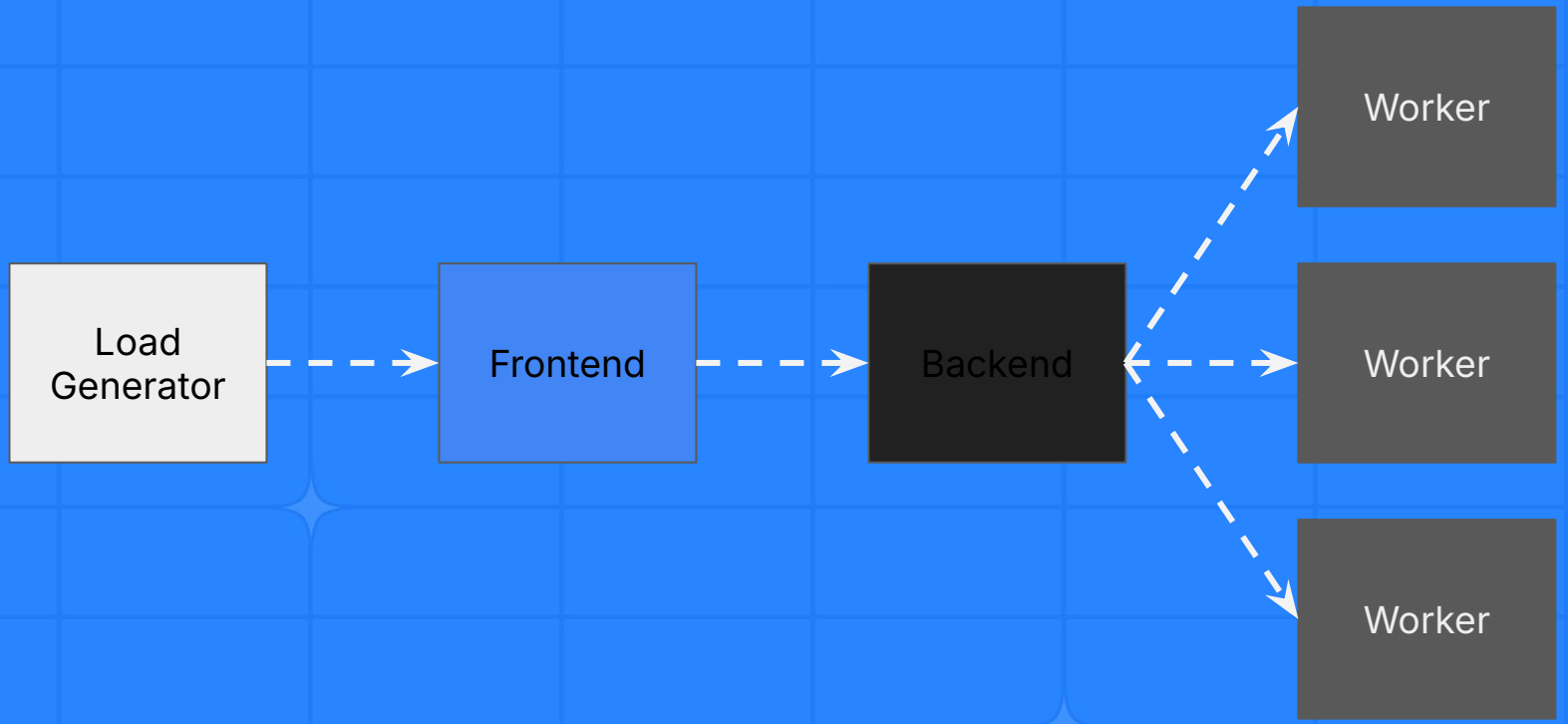
- u64 bpf_get_current_pid_tgid()
  - Returns the PID as seen from the Kernel (Namespace: 1) != PID as seen from Beyla
- struct task_struct* bpf_get_current_task()



The journey of an application trace



Demo time!



Minimal Beyla configuration

```
beyla-config.yml: |
  discovery:
    services:
      - k8s_namespace: demo
      - k8s_namespace: blog
```

Minimal Beyla configuration

```
- name: OTEL_EXPORTER_OTLP_ENDPOINT
  value: "https://otlp-gateway-prod-eu-west-0.grafana.net/otlp"
- name: OTEL_EXPORTER_OTLP_HEADERS
  valueFrom:
    secretKeyRef:
      key: otlp-headers
      name: grafana-secret
```

Search or jump to... ⌘+k

Home › Dashboards › Beyla RED M... Add Share Last 30 minutes 30s

Data source grafanacloud-mariomacias-prom Service All Job All Instance All

Slowest HTTP routes (P95)

service_name	http_route	Duration (ms)
backend	/factorial/*	241.375
frontend	/submit	241.37499999999997
frontend	/	4.75

Slowest RPC methods (P95)

rpc_method	service_name	Duration (ms)
/fib.Multiplier…	worker	241.15384615384

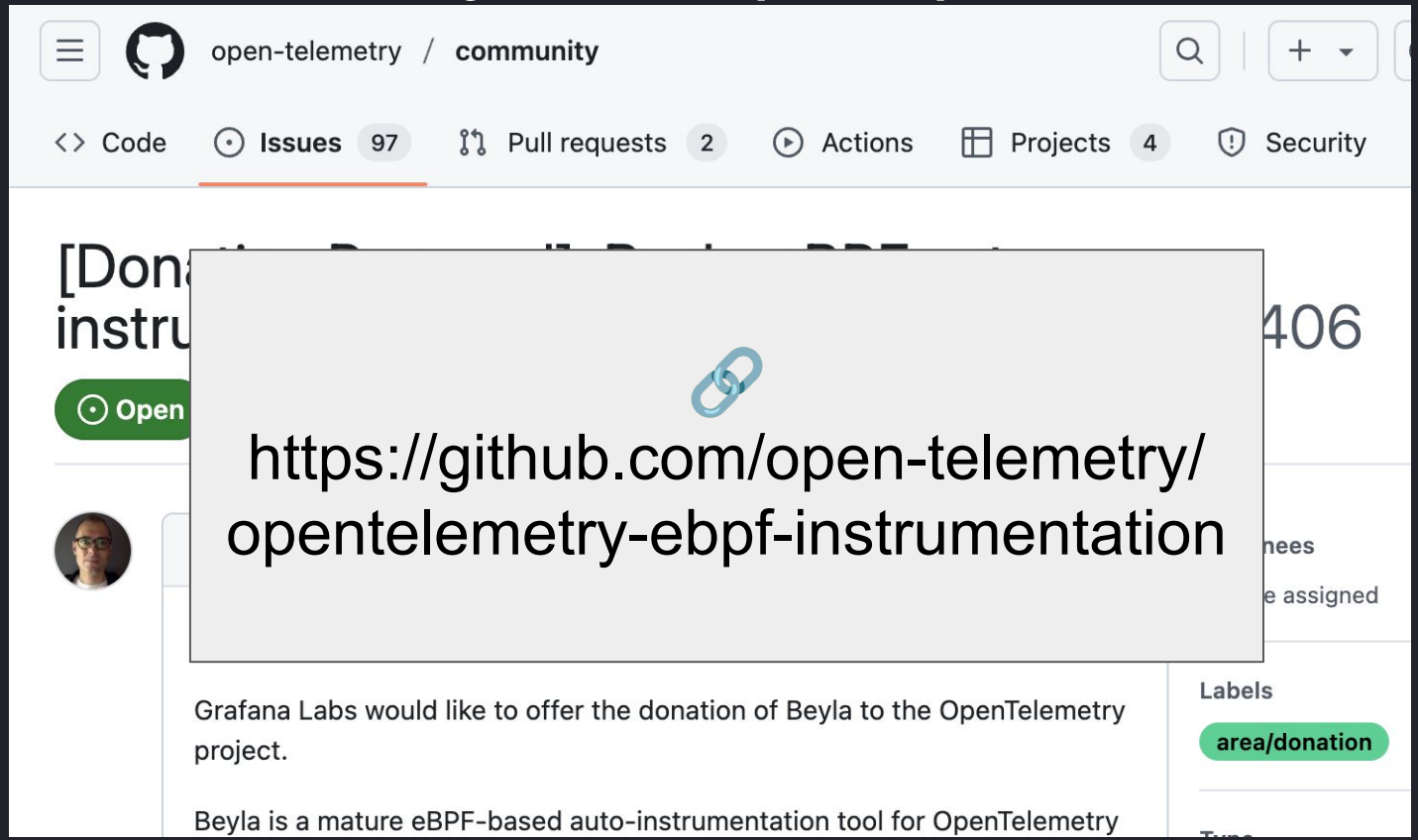
Inbound: backend

Duration

Request rate

Error rate

OSS Community: a core principle



Thank you and keep in touch!

Mario Macias
Staff Software Engineer

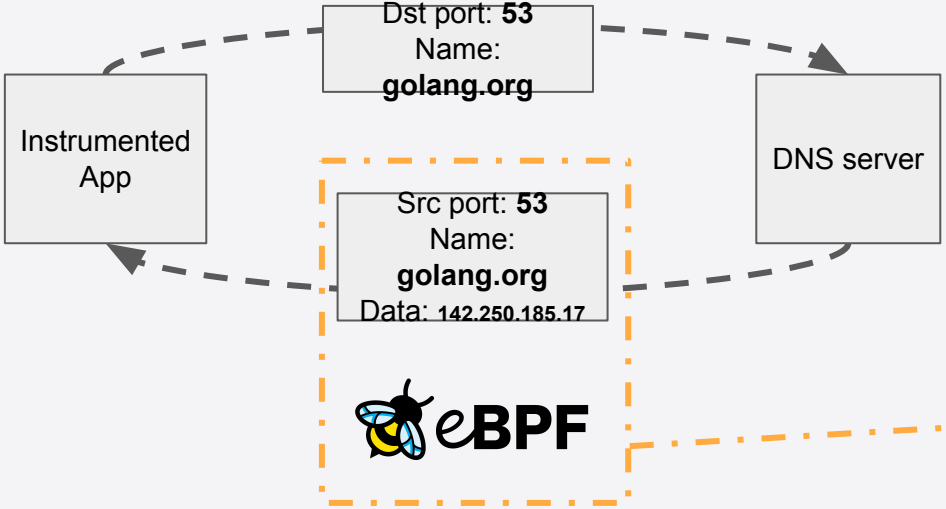
- Grafana Beyla
  - https://github.com/grafana/beyla
  - **#beyla** channel in grafana.slack.com

- Opentelemetry eBPF Instrumentation (OBI)
  - https://github.com/open-telemetry/opentelemetry-ebpf-instrumentation
  - **#otel-ebpf-instrumentation** channel in cloud-native.slack.com

Backup slides

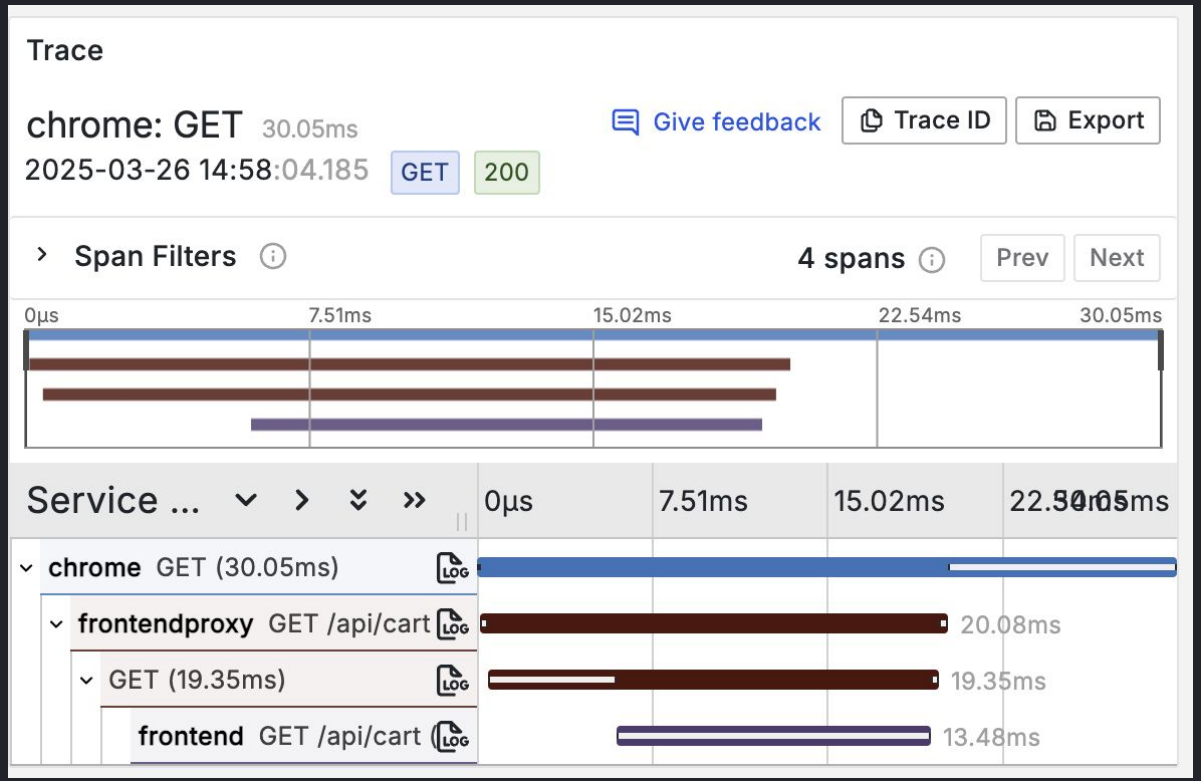
External traffic: Reverse DNS

```
$ ping -c 1 golang.org
PING golang.org (142.250.185.17): 56 data bytes
…
$ nslookup 142.250.185.17
…
17.185.250.142.in-addr.arpa name = mad41s11-in-f17.1e100.net.
```



network_flow_bytes{
…
dst_address=”142.250.185.17”,
dst_name=”golang.org”,
} 1357

Trace context propagation



HTTP

- Traceparent header

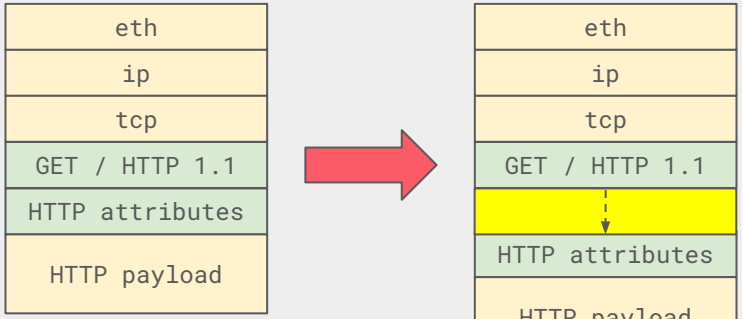
gRPC

- Metadata header

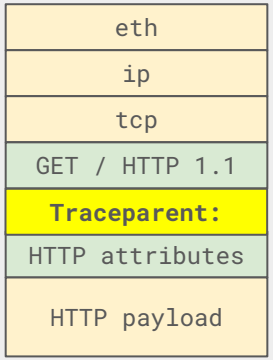
Instrumentation SDKs need to explicitly read it from inbound requests and inject it in outbound

HTTP context propagation

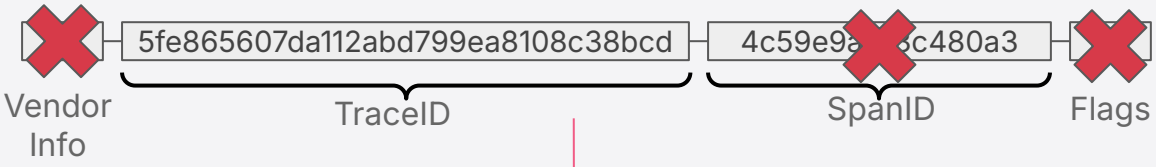
BPF_PROG_TYPE_SK_MSG



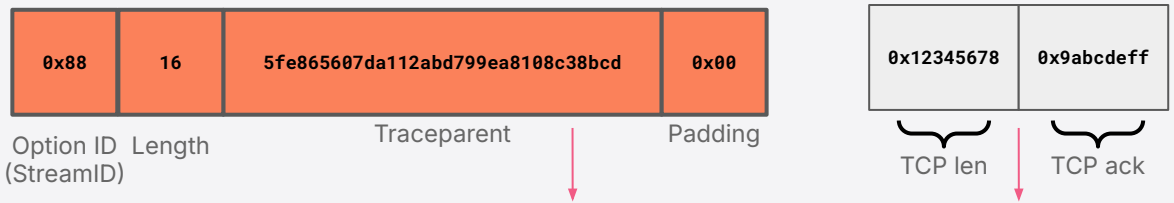
BPF_PROG_TYPE_SCHED_CLS



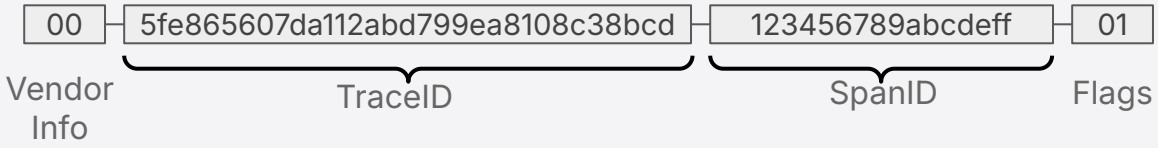
IP context propagation



Egress



Ingress



L5-L7: joining pieces

Time	Source	Event
	…	… previous stuff …
12	Socket (kernel)	Connection start (ports 53672→443)
13	Socket (kernel)	Connection start (ports 56380→8080)
15	Socket library	Response Body: “HTTP/1.1 200 OK\n ….”
17	TLS library	Request Body: “GET / HTTP/1.1\nHost: …”
17	HTTP library	Request Body: “POST /users/1234/product HTTP/1.1\nHost: …”
22	Socket library	Content: “(binary stuff) SELECT * FROM Users WHERE….”
23	Golang HTTP uprobes	Request Body: “GET / HTTP/1.1\nHost: …”
	…	… more stuff …

L5-L7: joining pieces

The easy way: classic web servers

Time	Thread ID	Parent Thread	Source	Payload
100	1	…	connect	src/dst address/port, etc…
110	123	1	sock_send	GET /users HTTP/1.1 Host: kube-service.ns User-Agent: …
115			accept	src/dst address/port, socket fd…
130	321	888	sock_recv	(binary stuff…)
143	123	1	sock_recv	HTTP/1.1 200 content-type: text/html; etc…
147	123	1	sock_recv	(more stuff…)
116	321	888	sock_send	(stuff…)
118	123	1	sock_close	

- Client-Side request
- HTTP protocol
- GET /users
- Request payload size
- Status 200 OK
- Response payload time
- Total transaction time

L5-L7: joining pieces

Playing in hell mode: modern async servers

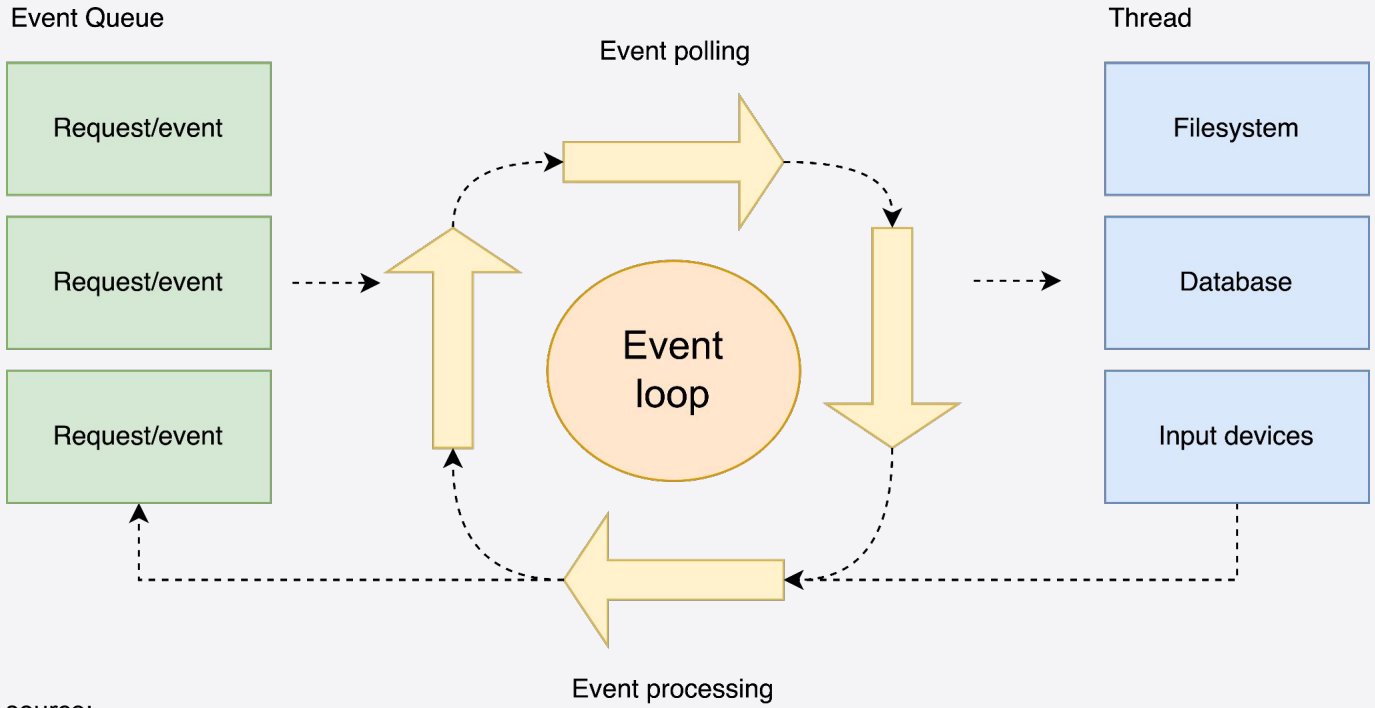


Image source:
https://tech.utugit.fi/soft/tools/lectures/dtek2054/2022/events/eventloop/

L5-L7: joining pieces

Playing in hell mode: modern async servers

- Need to
  - Go standard
    - Ma
  - Ne
    - Mai
  - Kafka
    - a tree